

Information Management for High Performance Autonomous Intelligent Systems

Scott Spetka
SUNY Institute of Technology
and ITT Corp.
Utica and Rome, NY, USA
scott@cs.sunyit.edu

Scot Tucker
ITT Corp.
775 Daedalian Drive
Rome, NY, USA
Scot.Tucker@itt.com

George Ramseyer
Richard Linderman
Air Force Research Laboratory
Rome, NY, USA
George.Ramseyer@rl.af.mil

Abstract— The publish/subscribe model for information management is particularly well suited for use in intelligent autonomous systems, ranging from robots to tactical communication systems. Information management systems that support pub/sub inherently provide a high degree of autonomy for users and communicating systems. The pub/sub paradigm can allow autonomous intelligent systems to communicate without requiring connection to a centralized brokering system. Each system is responsible for part of the overall brokering function, which imposes a cost for local system resources and proportionally diminishes the intelligence that can be expressed by each node. This raises the question of whether there exist controls that each intelligent autonomous system can use to avoid over-committing resources for publication brokering, such that node intelligence is uncompromised. Issues which affect autonomy in a pub/sub system that is currently under development are addressed.

Keywords: *Quality of Service (QoS), High Performance Computing (HPC), Autonomy, Broker, Pub/Sub, Intelligent*

I. INTRODUCTION

The main advantage of publish/subscribe (pub/sub) information management systems for autonomous intelligent systems is the decoupling of senders and receivers [1]. Instead of listening to particular publishers, subscribers can specify publications they want to receive by content, based on meta-data associated with publications. Similarly, publishers submit publications without regard to exactly which subscribers will receive them or whether they are currently listening for new publications. A *broker* performs the key function of matching publications with subscribers. Brokering depends on subscription information from end users (subscribers) and knowledge of structure for performing matching functions.

Optimal brokering for pub/sub information management systems that support quality of service (QoS) constraints requires simultaneously optimizing parameters that measure a range of criteria, including: bandwidth, latency, jitter and error rates. The problem is similar to the problem of optimal routing in a multicast system, except that routing is content-dependent for pub/sub systems. Because of the Nondeterministic Polynomial-time (NP) hard nature of the

problem, intelligent and heuristic approaches to routing for multi-constrained QoS multicast systems have been proposed [2][3].

The central issue for intelligent autonomous systems participating in a pub/sub brokering system while preserving a maximum degree of autonomy is the requirement that decisions be made based on a global state that can only be known through cooperation among participating brokers. But this places a requirement on the brokers to share their information and also to collect and maintain the information regarding remote systems that is needed locally. Requirements for storage, bandwidth and processing resources to support execution of intelligent algorithms and exchange of state information are generally proportional to a loss of autonomy due to participation in the system.

An intelligent autonomous pub/sub system is being developed at the Air Force Research Laboratory Information Directorate (AFRL/IF) [4]. Issues that affect autonomy and intelligence are surfacing in the system, and are being explored.

Implementing a distributed brokering service that scales well for increasing numbers of publications requires dynamically increasing resource usage as the number of publications being brokered increases. To meet QoS requirements for robustness, variable degrees of redundancy can be implemented. In addition, intelligent approaches to brokering must be considered, due to the complexity of the brokering problem in large systems and QoS constraints. Scalability for high performance information management systems provides the ability to add resources to handle increasing brokering loads on the system. Fairness issues must be considered and, when possible, measured, due to varying demands for resources to support cooperating brokers for pub/sub systems.

In the next section intelligent autonomous systems are introduced in the context of pub/sub information management systems. Then brokering architecture issues are discussed. The succeeding two sections present autonomy issues for this pub/sub architecture and for other architectures. The interplay

of intelligent brokering and autonomy is discussed for each approach. Related pub/sub applications that could also be implemented in a high performance computing (HPC) environment are described. Ideas for future research are presented, and consideration of whether scalable HPC pub/sub systems can support a high degree of autonomy for participating systems is presented in the conclusion.

II. INTELLIGENT AUTONOMOUS SYSTEMS

Intelligent autonomous pub/sub systems rely on brokering functions to match publications with subscribers (Figure 1). Some of the factors that can affect the performance of brokering include: buffer space, queued messages, message input rates, bandwidth among brokers and bandwidth between brokers and system end users (publishers and subscribers). Intelligent algorithms that manage brokering functions predict and plan for future processing requirements.

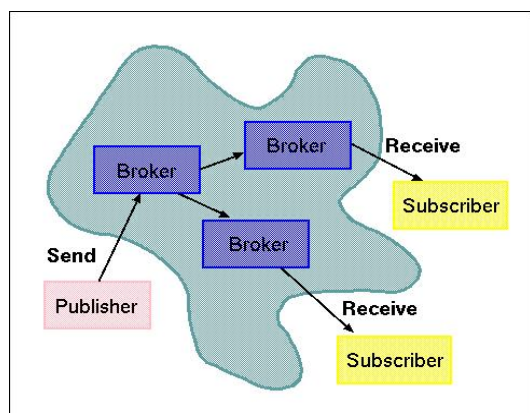


Fig. 1. Pub/Sub Information Management System

Intelligent systems are increasingly characterized by higher-level communication with understanding of content. Distributed system components receive inputs without specifying where those inputs should come from. Publications are sent without regard for the exact destinations. Some of the problems that system users face in formulating processing requests that are brokered by the system are similar to problems encountered when formulating requests to submit to an Internet search engine. For example, it may take several queries at a hardware store's Web site to find a water heater.

In a pub/sub-based system user inputs that specify a search for a local minimum or maximum on a surface could be published as a service request. There may be several subscribing services that could handle the request, depending on the degree of precision specified in the publication when the request is published. Results from each run can help the user to narrow a request, possibly by refining the required precision or by varying the search region.

In a pub/sub system, several subscriber HPC's that provide the requested service may receive the request and process it. In this case, several different responses may be received by the client that publishes the request, depending on the algorithm used for processing. It may be easy to choose the best result from the set of responses, eliminating the need for additional requests. After publishing a request for service, the user would normally wait for all processing sites to reply, to see if a good result has been returned, before sending in any further requests to refine the processing. However, lower latency can be achieved if an acceptable result is found, even if it is not optimal, so that processing can continue.

Our pub/sub architecture already implements a similar concept for reliable low-latency subscriptions. Subscribers always receive three subscriptions, through independent brokers. In this case, we know that all three will be identical, so we return the first publication received and ignore those that arrive later.

III. BROKERING ARCHITECTURE ISSUES

Our brokering architecture is designed to support the Joint Battlespace Infosphere (JBI) reference architecture [5]. The JBI specifies a common application programming interface (CAPI) for the interaction of end users, publishers and subscribers, with the system. The brokering function uses XML metadata, at least conceptually, to route publications from publishers to subscribers. As system load, measured by publications passing through the system, increases, demands on the brokering services increase. A parallel design provides scalability, which allows increasing the number of brokering nodes supporting the system.

An efficient pub/sub system that can operate across HPC systems is desirable, to allow load balancing and support processing for jobs that require more processors than may be available on any one HPC systems. Computations can also be distributed across hybrid HPC platforms when part of the computation may be performed more efficiently on particular architectures. For example, some parts of HPC codes perform better on shared memory systems, like the IBM P5, while other parts of the computation can take advantage of message passing on Linux clusters.

Resources that are contributed by a system to support distributed brokering activities on behalf of remote systems have the greatest impact on autonomy. Autonomous systems may be supporting brokering services even when there are no local publishers or subscribers. System performance will be degraded due to the support for other communicating systems that share the common pub/sub infrastructure.

Intelligent brokering systems generally require increased distributed state information at finer granularity, leading to increased storage, bandwidth and processing costs for each broker. In parallel broker designs, increased load can cause additional brokers to be dynamically added to the system. The HPC broker, implemented on a cluster computer, provides a capability for offloading processing, thereby enhancing autonomy for brokers and improving QoS processing.

IV. AUTONOMY IN HPC BROKER IMPLEMENTATIONS

Autonomy for brokers can be measured in terms of local storage, bandwidth and processing costs demanded of participating systems and also the degree to which individual systems can control their own resources. In an intelligent brokering system, cooperating brokers can offload work to other brokers, thereby improving overall system performance. However, forcing work on a broker may impact its ability to meet agreed upon QoS requirements. Of course, if the group of brokers as a whole agreed to a request for QoS, it would less affect the reputation of the underperforming broker. But, cooperative negotiations limit autonomy, by moving the decision to support a level of QoS for a publisher/subscriber from an individual broker to a committee.

The main advantage of our HPC brokering system is scalability. Within each HPC in our pub/sub environment, processing nodes may be dedicated to either brokering or other HPC applications. When additional brokering nodes are needed, due to increasing demands, in order to meet QoS requirements, they can be added at the HPC where the additional load will be supported. The decision to assign the load to a particular HPC, and whether the assigned processing load must be accepted, certainly impacts the autonomy of the system.

If HPCs make local decisions to voluntarily add brokering resources to the local broker pool, other HPCs could maintain smaller pools of broker nodes, giving them an unfair advantage. However, adding additional intelligence into decisions to increase the number of broker nodes increases overhead and can ultimately lead to committee decisions to allocate additional brokers at a given HPC, again resulting in the erosion of autonomy for the HPC which must contribute resources.

Defining and measuring autonomy for brokers in an intelligent Pub/Sub system is the key to providing QoS controls and assurance. In our HPC pub/sub implementation, increased communication requirements are gracefully supported by gradually reducing available processing resources to maintain an appropriate level of communications support for applications where processing is distributed across HPC systems. Figure 2 shows four HPC centers sharing

resources to provide an execution environment for three parallel programs. One of the programs is performing digital signal processing, another is performing cryptanalysis and another is executing the Modtran atmospheric analysis program. All three applications depend on the pub/sub system, which is shown spanning all four HPC's, for their communications needs. Each of the HPC centers is making some processors available for use by the pub/sub system in supporting system-wide communications.

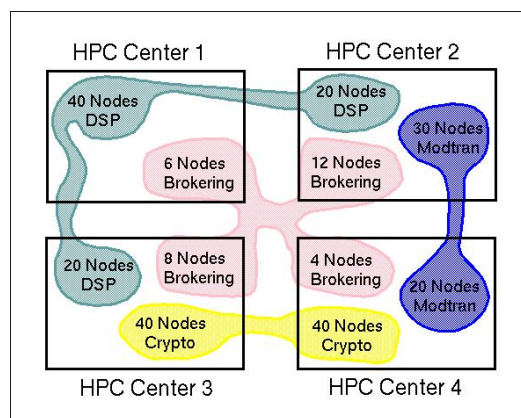


Fig. 2. Distributed Broker Architecture for HPC

V. AUTONOMY IN OTHER BROKER IMPLEMENTATIONS

Peer-to-peer networks can be used to implement pub/sub, but they naturally infringe upon the autonomy of workstations participating in distribution of messages. Increased activity for brokering on behalf of publication streams that pass through a peer system which is neither their origination nor their destination impose a load that may not be particularly welcome. The more intelligent the brokering scheme, the more processing and storage overhead are imposed on the cooperating peer. Similar concerns for autonomy have been studied for mobile peer-to-peer networks [6].

Serving as a broker for an open peer-to-peer system could also have implications for autonomy such as the loss of ability to filter messages based on content. Administrators may be responsible for transporting messages without ever approving of the users sending them or of the types of messages that they are sending. The Freenet [7] is an example of a dissemination system that is not exactly a pub/sub system. Participating Freenet sites must relinquish some of their control, especially over content. In the Freenet, "Users contribute to the network by giving bandwidth and a portion of their hard drive (called the 'data store') for storing files." Part of the mechanism which ensures the privacy of Freenet users is based on encrypting messages that are routed through the Freenet.

Some distributed broker architectures implement agent-based approaches. These approaches usually assume that agents can be decoupled from the entities that they represent. However, as in the peer-to-peer case, increases in processing, storage and communication, associated with increasingly intelligent algorithms, reduce the autonomy of participating systems that support brokering functions. Enhancing autonomy for perceptive middleware and intelligent agents is considered by Dimakic [8].

VI. RELATED WORK

There is a lot of work on QoS in pub/sub systems, but most of it pays little attention to autonomy issues. The SIENA publish/subscribe event notification service [9] is dynamically reconfigurable to adapt to the processing requirements of brokers using feedback from the on-line evaluation of performance models. SIENA routers can be dynamically added when they are needed, and routing functions can be redistributed. The idea is similar to our approach to scaling, explained above.

The Object Management Group (OMG) [10] Distributed Data Service for Real-Time Systems (DDS) standard [11] is an open international middleware standard directly addressing publish-subscribe communications for real-time and embedded systems. The DDS standard has been partially implemented with The Ace Orb (TAO) by several companies, including; Object Computing, Inc. [12], Real-Time Innovations, Inc. [13], Prism Technologies, Inc. [14]. While autonomy is not a primary consideration for DDS, it places content filtering functions close to the end uses and brokers based on "topics". Users subscribe and publish to topics. Brokering topics minimizes the need for intelligent brokering, but increases communication costs for publications in topics which are filtered when they arrive at the subscriber.

VII. FUTURE RESEARCH

Distributed architectures afford the opportunity to assign brokering for incoming subscriptions fairly among participating brokers. In systems where acceleration techniques are used to enhance brokering services, it may be both fair and efficient to concentrate new subscriptions for implementation at a single broker, but assign batches of new subscriptions to brokers in a round robin manner.

This approach would be effective in systems where field-programmable gate arrays (FPGAs) are used to support brokering. Since it is expensive to synthesize and load a new FPGA design, the cost should be shared evenly among all brokers. It should have a minimal overall effect on publication throughput rates during update cycles, when enough new

subscriptions have been received to warrant the cost of rebuilding the FPGA.

Over a longer time frame, our intelligent autonomous pub/sub-based system will need to implement a new paradigm for distributed computing that goes beyond SOAP [15] and Grid [16] protocols currently implemented for distributed computing. All routing in our system will intelligently find dynamically changing destinations for services that may help to find a solution to a problem, similar to the way that humans solve problems today.

VII. CONCLUSION

Our HPC cluster broker architecture shows that autonomy and scalability share similar characteristics, making scalable HPC architectures appear to be a good approach to implement autonomous pub/sub information management systems. The more brokers we have, the less they have to cooperate. In general, when functions are bound to particular locations, it limits autonomy by making it difficult to decide locally that a service should migrate to another system, to recover local resources. Scalability assures that additional processing resources can be used effectively and that applications are designed with component granularity that supports component migration.

We have shown that approaches to autonomy are feasible for pub/sub information management systems. More intelligence requires more knowledge of what is happening at remote brokers, loads on specific queues, etc. Scaling the brokering support provides the needed resources to support increasing intelligence in systems. Although this architecture is proven for general-purpose information management systems, we believe that it is well suited to support information management functions in other areas of autonomous intelligent distributed systems as well.

REFERENCES

- [1] Combs, V., Linderman, M., "A Jini-Based Publish and Subscribe Capability", Proceedings of SPIE -- Volume 4863, Java/Jini Technologies and High-Performance Pervasive Computing, June 2002, pp. 59-69.
- [2] Yuan, Xin; Liu, Xingming; "Heuristic Algorithms for Multi-constrained Quality of Service Routing" INFOCOM 2001. Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE Volume 2, 22-26 April 2001 Page(s):844 - 853 vol.2
- [3] Wang, Junwei; Wang, Xingwei; Huang, Min, "Hybrid Intelligent QoS Multicast Routing Algorithm in NGI", Proceedings of the Sixth International Conference on Parallel and Distributed Computing, Applications and Technologies (PDCAT'05)
- [4] Madhavan, R., Messina, I., E., "Performance Metrics for Intelligent Systems (PerMIS) 2006 Workshop: Summary and Review", Applied Imagery Pattern Recognition (AIPR) Workshop: Theory and Application of Model-based Image Analysis, Cosmos Club, Washington DC, October 11-13, 2006.

- [5] Linderman, M., Combs, V.T., Hillman, R.G., Muccio, M.T., McKeel, R.W., "Joint Battlespace Inforphere (JBI): Information Management in a Netcentric Environment, AFRL-IF-RS-TR-2006-178, May 2006.
- [6] Jari, Veijalainen, "Autonomy, Heterogeneity and Trust in Mobile P2P Environments", International Conference on Multimedia and Ubiquitous Engineering, 2007. MUE '07. April 2007 Page(s):41 - 47
- [7] The Freenet Project - <http://freenetproject.org/>
- [8] Dimakis, N.; Soldatos, J.; Polymenakos, L.; Schenk, M.; Pfirrmann, U.; Burkle, A.; "Perceptive Middleware and Intelligent Agents Enhancing Service Autonomy in Smart Spaces" IEEE/WIC/ACM International Conference on Intelligent Agent Technology, 2006. IAT '06. Dec. 2006 Page(s):276 - 283
- [9] Caporuscio, M., Di Marco, A., Inverardi, P., "Run-time Performance Management of the Siena publish/subscribe Middleware", Proceedings of the 5th international workshop on Software and performance WOSP '05, July 2005
- [10] Object Management Group - omg.org
- [11] Data-Distribution Service for Real-Time Systems (DDS) - <http://portals.omg.org/dds>
- [12] Object Computing, Inc., <http://www.ociweb.com/products/dds>
- [13] Real-Time Innovations, Inc., http://www.rti.com/products/data_distribution/dds_leader.html
- [14] Prism Technologies, Inc. <http://www.prismtechnologiesinc.com>
- [15] <http://www.w3.org/TR/soap12-part0/>
- [16] <http://www.ogf.org/>